The future of photovoltaic energy storage power stations

Unlike std::future, which is only moveable (so only one instance can refer to any particular asynchronous result), std::shared_future is copyable and multiple shared future objects ...

The class template std::future provides a mechanism to access the result of asynchronous operations: An asynchronous operation (created via std::async, std::packaged_task, ...

2) Move constructor. Constructs a std::future with the shared state of other using move semantics. After construction, other.valid() == false.

If the future is the result of a call to async that used lazy evaluation, this function returns immediately without waiting. The behavior is undefined if valid () is false before the call to this ...

In this session recorded at Unreal Fest Orlando 2025, Lee Rosario of Brunswick Corporation explains how he uses Twinmotion to present automotive, marine, and conceptual ...

The promise is the "push" end of the promise-future communication channel: the operation that stores a value in the shared state synchronizes-with (as defined in std::memory_order) ...

impl<F> Future for Box<F> where F: Unpin + Future + ?Sized, Boxed futures only implement the Future trait when the future inside the Box implements Unpin. Since your function ...

Blocks until the result becomes available. valid() == true after the call. The behavior is undefined if valid() == false before the call to this function.

In this case it does work. In general, it probably doesn"t. I"m wondering how this break in backwards compatibility should in general be navigated. Perhaps installing a previous version of ...

The get member function waits (by calling wait ()) until the shared state is ready, then retrieves the value stored in the shared state (if any). Right after calling this function, valid () is false. ...

Web: https://falconengineering.co.za